Classification of Camera Trap Images of Endangered Animals

Harun Harman, Mustafa Cen[1]

Abstract

Wildlife is one of the phenomena that reflects the ecological state and, in a sense, the health of the world and symbolizes the power and continuity of life. On the other hand, the number of species diversity is one of the factors that reflect the situation of wildlife. Unfortunately year by year, many species are becoming extinct or on the verge of extinction. For this reason, it is crucial to keep track of these endangered animals and providing them a healthy living space. In that purpose, classifying endangered animals from camera trap study is proposed. In this project, Categorical Naive Bayes model is applied with overall accuracy 0.58 and different pretrained neural network models are applied such as VGG-16 with accuracy 0.93, ResNET 50 with accuracy 0.99 etc., our developed CNN model has reached 0.95 accuracy and finally SVM classifier model has reached 0.91 overall accuracy.

1. Introduction

As time has passed, various factors like the growth of industries, a rise in population, people spreading out (leading to habitat loss), impacts of climate, and environmental pollution have placed numerous living species at risk of extinction. Recognizing the vital role that these species play in maintaining the natural balance, the looming threat of extinction poses significant challenges for the future. Seeking solutions to address this issue, our study focuses on developing models capable of detecting both near to extinct and endangered species. Employing mostly straightforward approaches, we crafted Categorical Naive Bayes model, utilizing a dataset comprising 35,641 camera trap images across 10 classes, sourced from different datasets. Through this research, we aim to contribute practical insights towards the conservation of diverse life forms in the face of ongoing environmental challenges.

[1].

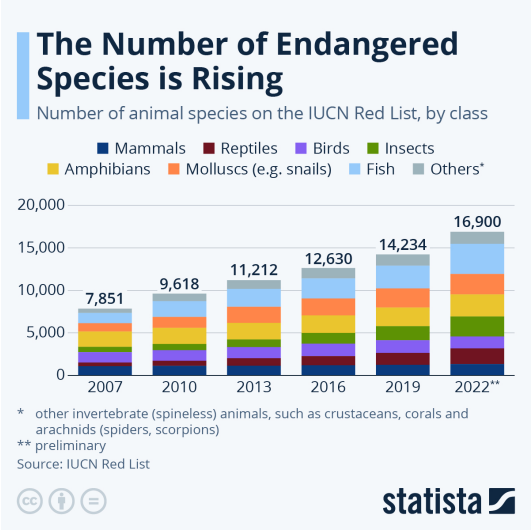

Figure 1. The Number of Endangered Animal Species Between 2007-2022

2. Related Work

Stanˇci´c et al. (2023) explored the efficiency of pretrained deep CNN models on camera trap images, demonstrating the strong performance of several architectures like ResNet and VGG-16.

On the other hand a quite different approach, Willi et al. (2019) took a pioneering step in blending citizen science and deep learning for camera trap image classification. Their work demonstrated how involving citizen volunteers can not only ease the burden of image annotation but also spark public interest and involvement in conservation efforts. While our own work dives deeper into model development and performance, it resonates with Willi et al.'s emphasis on making endangered species monitoring more accessible and engaging for the wider community.

Tan et al. (2023) explored the performance of various mainstream object detection architectures for animal detection and classification in camera trap images. Their evaluation of YOLOv5, Cascade R-CNN, and FCOS models aligns with the growing interest in applying these architectures to wildlife monitoring. Their findings, which demonstrated the effectiveness of certain architectures like YOLOv5 for fast and accurate detection, complement our own research

Figure 2. Number of samples in the train set.

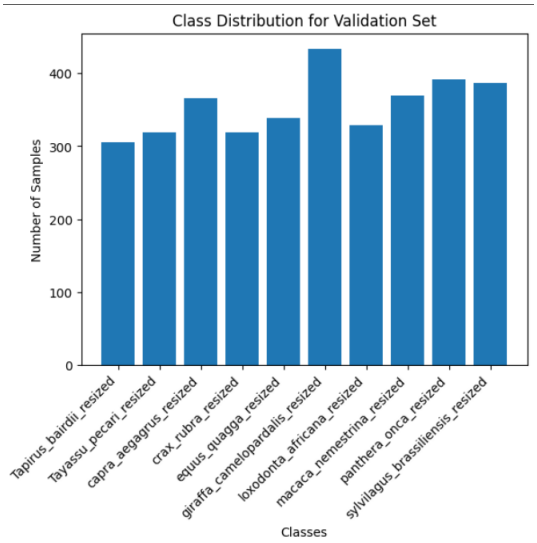

Figure 3. Number of samples in the validation set.

exploring model development and optimization for endangered animal classification.

3. The Approach

3.1. Data

3.1.1. Dataset

The dataset is constructed from two different datasets; Wcs and Orinoquia-camera-traps datasets which are taken from https://lila.science/datasets. It includes 10 different animal classes which are endangered, vulnerable or near to be extinct. We studied with 35641 images in total and resized it to different shapes such as (228,228), (64,64),(128,128). In above you can see the distribution of the classes in both train and validation sets.

Since the dataset is collected from different datasets and camera trap datasets are generally noisy, data collection part was quite challenging for us. We eliminated one by one from a total of 35641 data images that were unnecessary, distorted the structure, did not provide information about the animals, were repetitive, distorted, and had pixel problems, and this was a very time-consuming problem for the process.

3.1.2. Data Augmentation

After organizing the data, we observed various types of photos. Some were taken at night, some showed animals at a distance, and others had bursts of light. Additionally, there were photos where animals were too close to the camera, affecting the focus, and photos where only parts of the animals were visible due to their proximity. Alongside

Figure 4. Problematic images in datasets.

these, there were also a substantial number of more typical photos.

This diversity in the dataset poses a challenge for models to learn, but once they successfully adapt, it enables them to handle a wide range of situations.

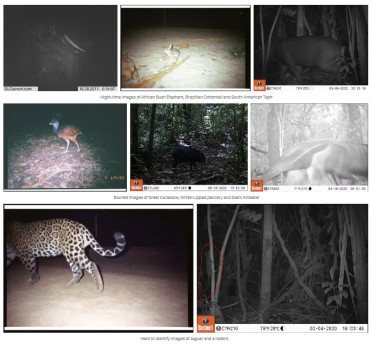

Figure 5. Various images includes various situations.

For augmentation, we used tensorflow's ImageDataGenerator.ImageDataGenerator is a class in TensorFlow's Keras API and it allows to do data augmentation in a fast way and reproduce the existing data and gather new images. In this way it enlarges the data and helps to get better accuracy. In our choice, we proceed with rotation, zooming, normalization of the pixels and apply preprocess function.

For example for only night images, we wanted to increase the brightness so that the animal can be more visible, in that order we wrote a preprocessing function for this and give it as a parameter in tensorflow's ImageDataGenerator library.

3.2. Model Methodology

3.2.1. Categorical Naive Bayes

The Categorical Naive Bayes algorithm is a probabilistic classification algorithm based on Bayes' theorem.This algorithm is an extension of the more general Naive Bayes algorithm, which assumes independence between features. The goal is to assign an instance to a predefined category or class based on its features.

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)}$$

Figure 6. Bayes Theorem.

In Figure 6, P(A) denotes the prior probability of each class.P(B|A) denotes the likelihood. This involves calculating the probability of each feature's occurrence within the given class. On the other hand P(A|B) denotes the posterior probability of each class given the observed features.

After having the probability for each class, the class which has the highest probability is assigned to the data.

3.2.2. Support Vector Classifier

The Support Vector Classifier (SVC) is a supervised machine learning algorithm used for classification tasks. It belongs to the family of Support Vector Machines (SVMs) and is effective in both linear and non-linear classification.

In our dataset, like many real-world scenarios, data is not linearly separable, meaning a single straight line (hyperplane) cannot effectively separate instances of different classes. So kernel trick is used to handle this issue.The kernel trick is a technique employed by SVC to map the input features into a higher-dimensional space, allowing for the possibility of finding a hyperplane that can effectively separate non-linearly separable data. A kernel function, denoted as K(a,b), computes the dot product of the transformed feature vectors in the higher-dimensional space without explicitly calculating the transformation. We used "radial basis kernel function" for our data because it performs better on non - linear classification. The formula of radial basis kernel function is:

$$K(a,b) = exp(-\frac{||a-b||^2}{2\gamma^2})$$

where gamma is an hyperparameter used in scikit-learn's SVC implementation.

3.2.3. Convolutional Neural Networks (CNN)

Convolutional Neural Networks are deep learning algorithms which is a subset of machine learning and specifically used for image classification tasks. The Convolutional Neural Network (CNN) stands as a formidable architecture in computer vision, characterized by three integral layers: the convolutional layer, pooling layer, and fully connected (FC) layer. The convolutional layer employs kernels or filters to traverse the image's receptive fields, systematically identifying features. This intricate convolution process lays the foundation for the network's ability to discern complex patterns within images. Subsequently, the pooling layer condenses and distills the information extracted, facilitating computational efficiency. The final FC layer integrates these hierarchical features, allowing the CNN to make informed predictions or classifications. In essence, the CNN's sequential progression through these layers mirrors the hierarchical nature of visual processing, rendering it a potent tool for tasks like image recognition in the realm of artificial intelligence.

4. Experimental Results

4.1. Categorical Naive Bayes:

In our camera trap image classification experiment, we employed a Categorical Naïve Bayes model, strategically processing the dataset in batches of 128 photographs to ensure efficient memory usage and prevent excessive strain on the RAM.

To identify the most effective feature extraction method for our data, we experimented with various approaches. One successful method involved flattening the RGB images and combining them with extracted Histogram of Oriented Gradients (HOG) features. However, this approach demanded a significant amount of RAM, especially when working with images of size (256,256,3) and a batch size of 16, ultimately leading to model crashes.

In an effort to optimize both computational resources and classification accuracy, we explored alternative feature extraction methods. We tested flattening Grayscale images and combining them with HOG features, as well as extracting edges using OpenCV's adaptiveThreshold method and then flattening the resulting edges image. Despite these attempts, the most promising results were achieved with RGB images of size (128,128,3) flattened and combined with HOG features.

This methodology allowed us to effectively leverage the Categorical Naïve Bayes model for classifying camera trap images, striking a balance between computational efficiency and classification accuracy. We measure the overall accuracy for naive bayes as 0.58 and below, you can see the class-wise metrics.

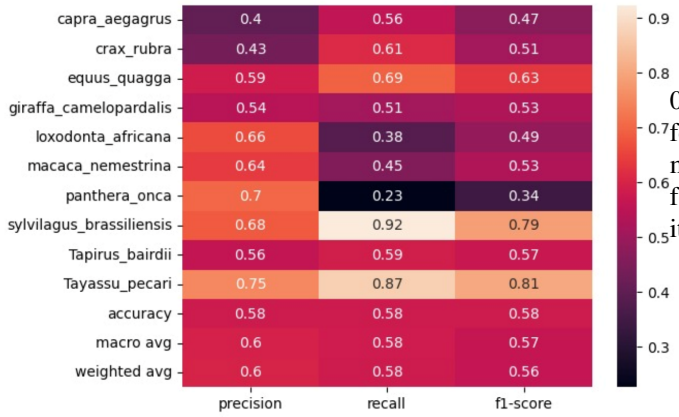

Figure 7. Example Confusion Matrix for Class-wise metrics.

Input Size	Precision	F1-Score	Accuracy
(64,64)	0.60	0.57	0.58
(128,128)	0.60	0.57	0.57
(256,256)	0.35	0.34	0.37

Table 1. Classification metrics for Categorical Naive Bayes.

4.2. Support Vector Machine Classifier:

With a more advanced model,SVM, we aimed to get better results compared to Naive Bayes. With this purpose, we start the process by first investigating the data and the data distribution.

Firstly, we extract the HoG features of each images and visualize them to decide if the data is linearly seperable or not.

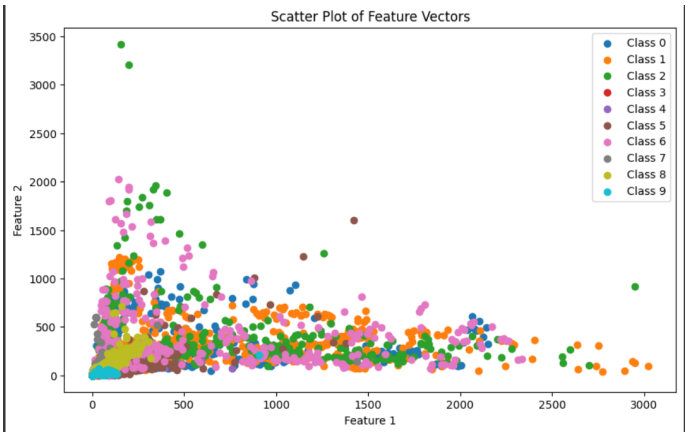

Figure 8. Scatter plot of the HoG features of the dataset.

As can be seen, data is non-linearly seperable and complex. To overcome this issue, we decided to proceed with the kernel function "rbf (radial basis function)".

Then, we applied train-test split to the data with the rate 0.2. So we have 28.512 train images and 7123 test images for the model. After the data preparation, we constructed the models. We applied GridSearchCV to find the best model for the problem and with the following parameters, we run it.

C Value	Gamma	Kernel	Score
1	1	RBF	0.124
10	10	RBF	0.124
100	100	RBF	0.124

Table 2. Parameter Selection for GridSearchCV

In Table 2, you can see that the score is too low and no improvement is gathered although the parameters are changed. Since we have limited hardware running GridSearchCV is time consuming and other related problems are occured, we could not do more experiments. Therefore we decided to proceed with "auto" gamma value and run the SVC models. This time you can see the results in Table 3.

C Value	Gamma	Kernel	Accuracy	F1-Score
1	Auto	rbf	0.80	0.80
10	Auto	rbf	0.88	0.88
100	Auto	rbf	0.91	0.91

Table 3. Parameter Selection for SVC

As illustrated above, the results are satisfying for our problem, but we will also check CNN approach and effects of it.

4.3. Convolutional Neural Networks

We proceed With a more popular and robust models which are CNN's. CNN's can be specialized for the data, which is a very nice feature. To reach this goal, we did lots of experiments with many hyperparameter tuning and found satisfying results. In this section,a model is developad and also "transfer learning" approach is applied and it will be mentioned in this section.

4.3.1. Implemented CNN Model

In the first place, we implemented our own model using "tensorflow" framework and "keras" library. In figure 9, you can see the architecture of our model.

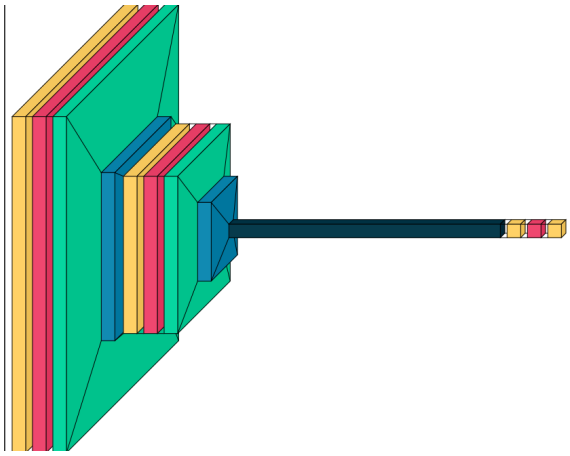

Figure 9. Architecture of CNN Model

In this model, we put 2 convolutional layers with filter number of 32,16 and 2 MaxPooling layers with size (2,2). After feeding the output to the flattening layer, the output enters a fully connected layer and finally the output layer with "softmax" activation function. In each hidden layer, "ReLu" activation function is preffered and gave the best result.

To prevent overfitting, some regularization techniques such as: "Batch normalization", "dropout layer addition (0.5)", "early stopping", "learning rate reduction" are applied. Because in the experiment phase, overfitting is observed and overcome with this approaches.

After preparing the model, we splitted data into train-validation-test with the rates 0.8,0.1,0.1. So, there were 28509 train, 3560 validation and 3572 test images. As a result in the test phase, the model reached 0.95 with 0.00001 learning rate.

4.3.2. VGG-16

We started trying several pretrained models with extra layers, we decided to try out VGG-16 Model first. While implementing the model, we wanted to stick with VGG-16's default input size which is (224,224,3), and it's pretrained layers. To prevent training pretrained layers, we froze those layers and added 3 layers on top of pretrained layers for higher accuracy. These layers are: Flattening layer to prepare the output for the fully connected layers, dense layer with 256 units and ReLu activation function, dropout regularization which randomly eliminates 50% of neurons, dense layer with 10 units and softmax activation function as output layer. After training the layers we added, we unfroze the pretrained layers and fine-tuned the model with a very low learning rate (0.0001). With this approach, we have managed to increase models' accuracy from 89% to 93%.

4.3.3. ResNET50

Another pretrained model is ResNet50, which is a strong model for image classification. The difference of the ResNet50 is the usage of the residual blocks in the architecture. These blocks helps solving the problem of vanishing gradients at deeper layers. Instead of trying to directly learn the desired mapping (output) for a set of input data, the model learns the difference or "residual" between the input and the desired output.

In our experiments, only the last FC layers of the ResNet50 are not included in the model and instead, 2 FC layers are added by us. Again, regularization techniques are applied. Since these models are too advanced for our problem and dataset, low learning rates are preferred (0.00005). According to the prediction results, overall accuracy is 0.99 which is great.

4.3.4. EFFICIENTNETB7

The last pretrained model is EfficientNetB7 which aims, as you can understand from its name, to achieve better performance while being computationally efficient. It applies scaling which involves increasing both the depth and width of the network in a balanced way. We set the same parameters for the model and get over 0.99 accuracy overall which is again great.

5. Conclusion

MODEL	LEARNING RATE	KERNEL	EPOCH	C VALUE	INPUT SIZE	PRECISION	RECALL	ACCURACY
CATEGORICAL NAIVE BAYES	-	-	-	-	(64,64,3)	0.60	0.58	0.58
SVM CLASSIFIER	-	RBF	-	100	(64,64,3)	0.91	0.91	0.91
IMPLEMENTED CNN	0.00001	CONVOLUTIONAL KERNEL (5,5)	20	-	(128,128,3)	0.95	0.95	0.95
VGG-16	0.00005	CONVOLUTIONAL KERNEL (3,3)	20	-	(224,224,3)	0.94	0.93	0.93
RESNET50	0.00005	CONVOLUTIONAL KERNEL (3,3)	20	-	(128,128,3)	0.99	0.99	0.99
EFFICIENTNETB7	0.00005	CONVOLUTIONAL KERNEL (1,1) AND (3,3)	20	-	(64,64,3)	0.99	0.99	0.99

Table 4. Model Comparison Table

As can be seen from the table when we compare the overall evaluation metrics ResNet50 and EfficientNetB7 have great performance for the problem. Since they are advanced networks, it was an expected result. Still, SVM, implemented CNN and VGG has important level of high accuracy and they are usable. On the other hand, naive bayes approach is not reliable and do not have the demanded level of accuracy.

From another perspective, computational cost, we faced with time-consuming experiments. Especially in transfer learning models and SVM, due to the relatively large dataset and the limit of the hardware and connection quality, we dealt with many problems and the limitation also limits the experiments.(For example larger range of parameter trials)

For future work, this project can be extended with larger datasets. Because for a machine learning project, data collection and getting the best data which reflects every possible scenario. In this way, real life applications are more reliable.

References

[1] Badhe, Tanishka Borde, Janhavi Waghmare, Bhagyashree Thakur, Vaishnavi Chaudhari, Anagha. (2022). Study of Deep Learning Algorithms to Identify and Detect Endangered Species of Animals.

[2] Stanˇci´c, A.; Vyroubal, V.; Slijepˇcevi´c, V. Classification Efficiency of Pre-Trained Deep CNN Models on Camera Trap Images. J. Imaging 2022, 8, 20. https://doi.org/10.3390/ jimaging8020020

[3] Willi M, Pitman RT, Cardoso AW, et al. Identifying animal species in camera trap images using deep learning and citizen science. Methods Ecol Evol. 2019; 10: 80–91. https://doi.org/10.1111/2041-210X.13099

[4] Tan, M.; Chao, W.; Cheng, J.-K.; Zhou, M.; Ma, Y.; Jiang, X.; Ge, J.; Yu, L.; Feng, L. Animal Detection and Classification from Camera Trap Images Using Different Mainstream Object Detection Architectures. Animals 2022, 12, 1976. https://doi.org/10.3390/ ani12151976

[5] Vélez J, McShea W, Shamon H, Castiblanco-Camacho PJ, Tabak MA, Chalmers C, Fergus P, Fieberg J. An evaluation of platforms for processing camera-trap data using artificial intelligence. Methods in Ecology and Evolution. 2023 Feb;14(2):459-77.